JAVA.UTIL.ARRAYS CLASS

Introduction

The **java.util.Arrays** class contains a static factory that allows arrays to be viewed as lists.Following are the important points about Arrays:

- This class contains various methods for manipulating arrays (such as sorting and searching).
- The methods in this class throw a NullPointerException if the specified array reference is null.

Class declaration

Following is the declaration for **java.util.Arrays** class:

```
public class Arrays
    extends Object
```

Class methods

S.N.	Method & Description
1	static <T> List<T> asList(T... a) This method returns a fixed-size list backed by the specified array.
2	static int binarySearch(byte[] a, byte key) This method searches the specified array of bytes for the specified value using the binary search algorithm.
3	static int binarySearch(byte[] a, int fromIndex, int toIndex, byte key) This method searches a range of the specified array of bytes for the specified value using the binary search algorithm.
4	static int binarySearch(char[] a, char key) This method searches the specified array of chars for the specified value using the binary search algorithm.
5	static int binarySearch(char[] a, int fromIndex, int toIndex, char key) This method searches a range of the specified array of chars for the specified value using the binary search algorithm.
6	static int binarySearch(double[] a, double key) This method searches the specified array of doubles for the specified value using the binary search algorithm.
7	static int binarySearch(double[] a, int fromIndex, int toIndex, double key) This method searches a range of the specified array of doubles for the specified value using the binary search algorithm.
8	static int binarySearch(float[] a, float key) This method searches the specified array of floats for the specified value using the binary search algorithm.
9	static int binarySearch(float[] a, int fromIndex, int toIndex, float key) This method searches a range of the specified array of floats for the specified value using the binary search algorithm.

10	static int binarySearch(int[] a, int key) This method searches the specified array of ints for the specified value using the binary search algorithm.
11	static int binarySearch(int[] a, int fromIndex, int toIndex, int key) This method searches a range of the specified array of ints for the specified value using the binary search algorithm.
12	static int binarySearch(long[] a, int fromIndex, int toIndex, long key) This method searches a range of the specified array of longs for the specified value using the binary search algorithm.
13	static int binarySearch(long[] a, long key) This method searches the specified array of longs for the specified value using the binary search algorithm.
14	static int binarySearch(Object[] a, int fromIndex, int toIndex, Object key) This method searches a range of the specified array for the specified object using the binary search algorithm.
15	static int binarySearch(Object[] a, Object key) This method searches the specified array for the specified object using the binary search algorithm.
16	static int binarySearch(short[] a, int fromIndex, int toIndex, short key) This method searches a range of the specified array of shorts for the specified value using the binary search algorithm.
17	static int binarySearch(short[] a, short key) This method searches the specified array of shorts for the specified value using the binary search algorithm.
18	static <T> int binarySearch(T[] a, int fromIndex, int toIndex, T key, Comparator<? super T> c) This method searches a range of the specified array for the specified object using the binary search algorithm.
19	static <T> int binarySearch(T[] a, T key, Comparator<? super T> c) This method searches the specified array for the specified object using the binary search algorithm.
20	static boolean[] copyOf(boolean[] original, int newLength) This method copies the specified array, truncating or padding with false (if necessary) so the copy has the specified length.
21	static byte[] copyOf(byte[] original, int newLength) This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
22	static char[] copyOf(char[] original, int newLength) This method copies the specified array, truncating or padding with null characters (if necessary) so the copy has the specified length.
23	static double[] copyOf(double[] original, int newLength) This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
24	static float[] copyOf(float[] original, int newLength) This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
25	static int[] copyOf(int[] original, int newLength)

	This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
26	static long[] copyOf(long[] original, int newLength) This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
27	static short[] copyOf(short[] original, int newLength) This method copies the specified array, truncating or padding with zeros (if necessary) so the copy has the specified length.
28	static <T> T[] copyOf(T[] original, int newLength) This method copies the specified array, truncating or padding with nulls (if necessary) so the copy has the specified length.
29	static <T,U> T[] copyOf(U[] original, int newLength, Class<? extends T[]> newType) This method copies the specified array, truncating or padding with nulls (if necessary) so the copy has the specified length.
30	static boolean[] copyOfRange(boolean[] original, int from, int to) This method copies the specified range of the specified array into a new array.
31	static byte[] copyOfRange(byte[] original, int from, int to) This method copies the specified range of the specified array into a new array.
32	static char[] copyOfRange(char[] original, int from, int to) This method copies the specified range of the specified array into a new array.
33	static double[] copyOfRange(double[] original, int from, int to) This method copies the specified range of the specified array into a new array.
34	static float[] copyOfRange(float[] original, int from, int to) This method copies the specified range of the specified array into a new array.
35	static int[] copyOfRange(int[] original, int from, int to) This method copies the specified range of the specified array into a new array.
36	static long[] copyOfRange(long[] original, int from, int to) This method copies the specified range of the specified array into a new array.
37	static short[] copyOfRange(short[] original, int from, int to) This method copies the specified range of the specified array into a new array.
38	static <T> T[] copyOfRange(T[] original, int from, int to) This method copies the specified range of the specified array into a new array.
39	static <T,U> T[] copyOfRange(U[] original, int from, int to, Class<? extends T[]> newType) This method copies the specified range of the specified array into a new array.
40	static boolean deepEquals(Object[] a1, Object[] a2) This method returns true if the two specified arrays are deeply equal to one another.
41	static int deepHashCode(Object[] a) This method returns a hash code based on the "deep contents" of the specified array.
42	static String deepToString(Object[] a) This method returns a string representation of the "deep contents" of the specified array.

43	static boolean equals(boolean[] a, boolean[] a2) This method returns true if the two specified arrays of booleans are equal to one another.
44	static boolean equals(byte[] a, byte[] a2) This method returns true if the two specified arrays of bytes are equal to one another.
45	static boolean equals(char[] a, char[] a2) This method returns true if the two specified arrays of chars are equal to one another.
46	static boolean equals(double[] a, double[] a2) This method returns true if the two specified arrays of doubles are equal to one another.
47	static boolean equals(float[] a, float[] a2) This method returns true if the two specified arrays of floats are equal to one another.
48	static boolean equals(int[] a, int[] a2) This method returns true if the two specified arrays of ints are equal to one another.
49	static boolean equals(long[] a, long[] a2) This method returns true if the two specified arrays of longs are equal to one another.
50	static boolean equals(Object[] a, Object[] a2) This method returns true if the two specified arrays of Objects are equal to one another.
51	static boolean equals(short[] a, short[] a2) This method returns true if the two specified arrays of shorts are equal to one another.
52	static void fill(boolean[] a, boolean val) This method assigns the specified boolean value to each element of the specified array of booleans.
53	static void fill(boolean[] a, int fromIndex, int toIndex, boolean val) This method assigns the specified boolean value to each element of the specified range of the specified array of booleans.
54	static void fill(byte[] a, byte val) This method assigns the specified byte value to each element of the specified array of bytes.
55	static void fill(byte[] a, int fromIndex, int toIndex, byte val) This method assigns the specified byte value to each element of the specified range of the specified array of bytes.
56	static void fill(char[] a, char val) This method assigns the specified char value to each element of the specified array of chars.
57	static void fill(char[] a, int fromIndex, int toIndex, char val) This method assigns the specified char value to each element of the specified range of the specified array of chars.
58	static void fill(double[] a, double val) This method assigns the specified double value to each element of the specified array of doubles.
59	static void fill(double[] a, int fromIndex, int toIndex, double val) This method assigns the specified double value to each element of the specified range of the specified array of doubles.
60	static void fill(float[] a, float val) This method assigns the specified float value to each element of the specified array of floats.

61	static void fill(float[] a, int fromIndex, int toIndex, float val) This method assigns the specified float value to each element of the specified range of the specified array of floats.
62	static void fill(int[] a, int val) This method assigns the specified int value to each element of the specified array of ints.
63	static void fill(int[] a, int fromIndex, int toIndex, int val) This method assigns the specified int value to each element of the specified range of the specified array of ints.
64	static void fill(long[] a, int fromIndex, int toIndex, long val) This method assigns the specified long value to each element of the specified range of the specified array of longs.
65	static void fill(long[] a, long val) This method assigns the specified long value to each element of the specified array of longs.
66	static void fill(Object[] a, int fromIndex, int toIndex, Object val) This method assigns the specified Object reference to each element of the specified range of the specified array of Objects.
67	static void fill(Object[] a, Object val) This method assigns the specified Object reference to each element of the specified array of Objects.
68	static void fill(short[] a, int fromIndex, int toIndex, short val) This method assigns the specified short value to each element of the specified range of the specified array of shorts.
69	static void fill(short[] a, short val) This method assigns the specified short value to each element of the specified array of shorts.
70	static int hashCode(boolean[] a) This method returns a hash code based on the contents of the specified array.
71	static int hashCode(byte[] a) This method returns a hash code based on the contents of the specified array.
72	static int hashCode(char[] a) This method returns a hash code based on the contents of the specified array.
73	static int hashCode(double[] a) This method returns a hash code based on the contents of the specified array.
74	static int hashCode(float[] a) This method returns a hash code based on the contents of the specified array.
75	static int hashCode(int[] a) This method returns a hash code based on the contents of the specified array.
76	static int hashCode(long[] a) This method returns a hash code based on the contents of the specified array.
77	static int hashCode(Object[] a) This method returns a hash code based on the contents of the specified array.

78	static int hashCode(short[] a) This method returns a hash code based on the contents of the specified array.
79	static void sort(byte[] a) This method sorts the specified array of bytes into ascending numerical order.
80	static void sort(byte[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of bytes into ascending numerical order.
81	static void sort(char[] a) This method sorts the specified array of chars into ascending numerical order.
82	static void sort(char[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of chars into ascending numerical order.
83	static void sort(double[] a) This method sorts the specified array of doubles into ascending numerical order.
84	static void sort(double[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of doubles into ascending numerical order.
85	static void sort(float[] a) This method sorts the specified array of floats into ascending numerical order.
86	static void sort(float[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of floats into ascending numerical order.
87	static void sort(int[] a) This method sorts the specified array of ints into ascending numerical order.
88	static void sort(int[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of ints into ascending numerical order.
89	static void sort(long[] a) This method sorts the specified array of longs into ascending numerical order.
90	static void sort(long[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of longs into ascending numerical order.
91	static void sort(Object[] a) This method sorts the specified array of objects into ascending order, according to the natural ordering of its elements.
92	static void sort(Object[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of objects into ascending order, according to the natural ordering of its elements.
93	static void sort(short[] a) This method sorts the specified array of shorts into ascending numerical order.
94	static void sort(short[] a, int fromIndex, int toIndex) This method sorts the specified range of the specified array of shorts into ascending numerical order.
95	static <T> void sort(T[] a, Comparator<? super T> c) This method sorts the specified array of objects according to the order induced by the specified comparator.
96	static <T> void sort(T[] a, int fromIndex, int toIndex, Comparator<? super T> c)

	This method sorts the specified range of the specified array of objects according to the order induced by the specified comparator.
97	static String toString(boolean[] a) This method returns a string representation of the contents of the specified array of boolean.
98	static String toString(byte[] a) This method returns a string representation of the contents of the specified array of bytes.
99	static String toString(char[] a) This method returns a string representation of the contents of the specified array of chars.
100	static String toString(double[] a) This method returns a string representation of the contents of the specified array of doubles.
101	static String toString(float[] a) This method returns a string representation of the contents of the specified array of floats.
102	static String toString(int[] a) This method returns a string representation of the contents of the specified array of ints.
103	static String toString(long[] a) This method returns a string representation of the contents of the specified array of longs.
104	static String toString(Object[] a) This method returns a string representation of the contents of the specified array of ints.
105	static String toString(short[] a) This method returns a string representation of the contents of the specified array of shorts.

Methods inherited

This class inherits methods from the following classes:

- java.util.Object